Python Fun(damentals)

Down Selecting Packages

Alexander Rymdeko-Harvey

Obscurity Labs

```
* Mastering Python
* Avoiding reinventing the wheel
* Picking high-quality packages
```

Mastering Python != Mastering Coding

A couple of key things to keep in mind when we talk about building Python applications:

1. Understand the right workflows and tools of the “ecosystem” surrounding the core language.
2. Don't spend time writing common building blocks like config file parsers, data validators, and serializers.
3. Spend the time to truly know the workflow and logic behind programming theory. We call these *patterns*.
4. Don't reinvent the wheel, you are likely to do it wrong or miss critical logic.

Don't Reinvent the wheel!

A few notes to prevent this `trap` from happening and building confidence in open source code:

- Manage your overconfidence during the planning phase.
- Truly learn pip package management, virtual environments, and requirements files.
- You can build confidence in your package selection through various methods:
  - Test, test, test. If you don't test, your code is broken.
  - Pin your packages by versions!
- Write code at a higher level of abstraction to focus on the business need not the technical solution.
- If a solution does not exist READ blogs, posts and any ancillary information you can gather on the problem set to increase the likelihood of success.

Picking High-Quality Packages

Here are a few tips when selecting Python Packages:

- Make sure the package has adequate unit tests.
- Make sure the package has published code coverage (How much of the source is unit tested).
- Make sure the package is actively maintained:
  - Pull requests.
  - Issues are being resolved.
  - Recent releases or commits to the code base.
  - NO vulnerabilities or reported security concerns.
- Make sure the package has solid documentation and can achieve your goal in a clearly defined matter.
- Check to make sure the source control management is being done properly.